(19)日本国特許庁（ＪＰ）　(12) 公開特許公報（A）　(11)特許出願公開番号
特開2001－117769
（P2001－117769A）
(43)公開日　平成13年４月27日(2001.4.27)

(51)Int.Cl.'	識別記号	ＦＩ	ﾃｰﾏｺｰﾄﾞ(参考)
Ｇ０６Ｆ　9/06	５５０	Ｇ０６Ｆ　9/06	５５０Ｚ　５Ｂ０７６

審査請求　未請求　請求項の数８　ＯＬ　（全 17 頁）

(21)出願番号　特願平11－297759

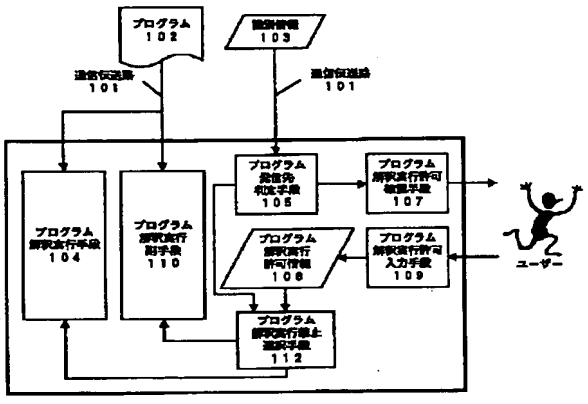
(22)出願日　平成11年10月20日(1999.10.20)

(71)出願人　000005821
松下電器産業株式会社
大阪府門真市大字門真1006番地
(72)発明者　金丸　智一
大阪府門真市大字門真1006番地　松下電器産業株式会社内
(74)代理人　100097445
弁理士　岩橋　文雄　（外２名）

Ｆターム(参考)　5B076 BB06 CA07 FD00

(54)【発明の名称】　プログラム実行装置

(57)【要約】
【課題】　プログラム配信のためのネットワーク機能を備えた家電機器において、ダウンロードしたプログラムの安全性・秘匿性・完全性の保証を、低コストかつ簡便に実現する。
【解決手段】　プログラム解釈実行手段１０４と、プログラム解釈実行副手段１１０と、プログラム発信先判定手段１０５と、プログラム解釈実行許可入力手段１０９と、プログラム解釈実行禁止選択手段１１２とから構成される。プログラム発信先判定手段１０５の判定結果が是である場合には、プログラム解釈実行禁止選択手段１１２は、プログラム解釈実行手段１０４がプログラム１０２を解釈実行することを許可し、判定結果が非である場合には、プログラム解釈実行禁止選択手段１１２は、プログラム解釈実行手段１０４がプログラム１０２を解釈実行することを禁止する。



プログラム実行装置・その４

1

【特許請求の範囲】

【請求項１】　外部伝送路を通して送信される、プログラムと外部伝送路を通して送信される、プログラムの発信元の識別情報とを入力とする、プログラム実行装置であり、

プログラム実行装置は、

プログラムを解釈実行する手段である、プログラム解釈実行手段と、

識別情報が、信頼されたプログラム発信元を示すものであるか否かを判定する手段である、プログラム発信先判定手段と、

プログラム解釈実行禁止手段と、

からなり、

プログラム発信先判定手段の判定に従って、プログラム解釈実行禁止手段は、プログラム解釈実行手段がプログラムを解釈実行するのを禁止することを特徴とする、プログラム実行装置。

【請求項２】　外部伝送路を通して送信される、プログラムと外部伝送路を通して送信される、プログラムの発信元の識別情報とを入力とする、プログラム実行装置であり、

プログラム実行装置は、

前記プログラム解釈実行手段と、

前記プログラム発信先判定手段と、

前記プログラム解釈実行禁止手段と、

プログラムの解釈実行を許可する、あるいは、禁止する、のいずれかを示す、解釈実行許可情報を、ユーザが入力する手段である、プログラム解釈実行許可入力手段と、

からなり、

解釈実行許可情報に従って、プログラム解釈実行禁止手段は、プログラム解釈実行手段がプログラムを解釈実行するのを禁止することを特徴とする、プログラム実行装置。

【請求項３】　外部伝送路を通して送信される、プログラムと外部伝送路を通して送信される、プログラムの発信元の識別情報とを入力とする、プログラム実行装置であり、

プログラム実行装置は、

前記プログラム解釈実行手段と、

プログラムを解釈実行する第２の手段である、プログラム解釈実行副手段と、

前記プログラム発信先判定手段と、

プログラム解釈実行選択手段と、

からなり、

プログラム発信先判定手段の判定に従ってプログラム解釈実行選択手段が、プログラムがプログラム解釈実行手段で実行されるのを許可するかあるいは禁止するかを選択して、プログラムがプログラム解釈実行手段で実行されるのを禁止する場合には、プログラムをプログラム解釈実行手段副手段で実行させることを特徴とする、プログラム実行装置。

2

【請求項４】　外部伝送路を通して送信される、プログラムと外部伝送路を通して送信される、プログラムの発信元の識別情報とを入力とする、プログラム実行装置であり、

プログラム実行装置は、

前記プログラム解釈実行手段と、

前記プログラム解釈実行副手段と、

前記プログラム発信先判定手段と、

プログラムの解釈実行を許可する、あるいは、禁止する、のいずれかを示す、前記解釈実行許可情報を、ユーザが入力する手段である、前記プログラム解釈実行許可入力手段と、

プログラム解釈実行禁止選択手段と、

からなり、

プログラム発信先判定手段の判定結果が是である場合には、プログラム解釈実行禁止選択手段は、プログラム解釈実行手段がプログラムを解釈実行することを許可し、

プログラム発信先判定手段の判定結果が非である場合には、プログラム解釈実行禁止選択手段は、プログラム解釈実行手段がプログラムを解釈実行することを禁止することを特徴とし、

プログラム発信先判定手段の判定結果が非である場合には、プログラム解釈実行禁止選択手段はは前記解釈実行許可情報に従って、プログラム解釈実行副手段がプログラムを解釈実行することを許可する、あるいは禁止する、のいずれかを選択することを特徴とする、プログラム実行装置。

【請求項５】　プログラム発信先判定手段の判定結果が否である場合、プログラムの解釈実行を許可するか否かをユーザに問い合わせる手段である、プログラム解釈実行許可確認手段を備えることを特徴とした、請求項２および請求項４のプログラム実行装置。

【請求項６】　識別情報はＩＰアドレスを含み、プログラム発信先判定手段は、ＩＰアドレスを用いて判定を行うことを特徴とする、請求項１、２、３、４、５記載のプログラム実行装置。

【請求項７】　識別情報はＵＲＬを含み、プログラム発信先判定手段１０５は、ＵＲＬを用いて判定を行うことを特徴とする、請求項１、２、３、４、５記載のプログラム実行装置。

【請求項８】　識別情報は暗号化された署名を含み、プログラム発信先判定手段は、暗号化された署名を用いて判定を行うことを特徴とする、請求項１、２、３、４、５記載のプログラム実行装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明はネットワーク通信によりプログラムをダウンロードし、プログラムを解釈実

行する環境、特に家電機器など、使用目的が特化されたプログラムを解釈実行する環境において、ダウンロードされたプログラムの検証を行なう装置およびその方法に関する。

【０００２】

【従来の技術】近年、家電機器に対するプログラム配信のためのネットワーク機能搭載の要求が高まっている。

【０００３】ネットワークを通じてプログラムの配信が行われることで、例えば、家電機器に対する機能の更新や追加が容易になり、新たなサービスの提供が可能になる、などの多くの利点が生まれる。

【０００４】既にコンピューティングの分野では、こういったプログラムの配信は頻繁に行なわれている。

【０００５】その代表的なものに、インターネットのウェブサーバと、ブラウザを載せたＰＣの間のネットワークがある。このネットワークにおいては、ネットワークに接続する者であれば誰でも、そこに流通するコンテンツを自由にダウンロードして利用できる。ここではこの性質を持つネットワークを「開かれたネットワーク」と呼ぶことにする。

【０００６】開かれたネットワークの元では、利用者はネットワークに存在するプログラムもまた、自由にダウンロードし、実行することができる。

【０００７】しかし一方で、開かれたネットワークでは、以下のことをネットワーク利用者によるリスクの分担の上で行なうという特徴を持つ。１．接続先が意図した相手であること（認証と許可）。２．コンテンツが伝送経路で改ざんや盗聴がなされないこと（完全性と秘匿性）。３．コンテンツがそれを利用する環境で危険な振る舞いをしないこと（安全性）。

【０００８】つまり開かれたネットワークとは、上記の１．２．３．が完全に保証される環境ではない。そのため、悪意ある第３者によって作成された危険なプログラムを、そうとは知らずに利用者がダウンロードし、実行してしまう可能性がある。

【０００９】一方、開かれたネットワークとは異なり、流通するコンテンツをダウンロードする際に、コンテンツの暗号化やコンテンツをダウンロードする際の利用者の認証情報の確認などの必要を課すことによって、上記の１．２．３．の点を保証するネットワークも存在する。この性質を持つネットワークを「閉じられたネットワーク」と呼ぶことにする。

【００１０】家電機器には一般的に高い信頼性・安全性が求められており、家電機器に対するプログラム配信の際には、信頼できる配信元が作成した安全なプログラムのみがダウンロードされることが求められる。

【００１１】加えて家電機器に対するプログラム配信には、しばしば課金やデータの保護などが安全・正確に行われることが必要である。これを実現するには開かれたネットワークでは不充分である。これらの観点から、家

電機器に対するコンテンツの配信は、閉じられたネットワークが使用されることが望ましい。

【００１２】実際、現行の家電機器向けを想定したネットワークは、何らかの手段によって上記の１．２．３．の点を保証した、閉じられたネットワークであるものが殆どである。

【００１３】例えば現行の電話を用いた各種のサービスも、電話のネットワークを利用し、通話によるネットワークの利用に伴う課金を正確に行なう必然性から、そのサービスは閉じられたネットワークで実施されている。もちろん、コンテンツに暗号化等の処理を加えることにより、より強固で安全なネットワークとすることも可能である。

【００１４】

【発明が解決しようとする課題】しかしながら近年、家電機器分野においても、インターネットに接続可能な携帯電話などのサービスに代表されるように、開かれたネットワーク環境に接続することが、一部の機器では可能になりつつある。インターネットに代表される開かれたネットワークと、家電機器との親和性は、今後さらに高まっていくであろうことが予想される。

【００１５】このような状況においては先に述べたように、閉じられたネットワークを用い、信頼できる発信元からのみ、保証されたプログラムをダウンロードし実行することを、家電機器に遵守させることは難しくなる。

【００１６】一つの解決策は、プログラムを解釈実行する実行環境のほうに、プログラムが危険なものでないかどうかを検査する検証機構を持たせる、というものである。

【００１７】この解決策をとっている代表的なアーキテクチャにはＪａｖａがある。

【００１８】開発言語としてのＪａｖａ言語は、オブジェクト指向言語としての開発効率の高さ、バグが出難い言語仕様、移植性の容易さ、既存の開発環境の豊富さ、など多くの優れた点を持っている。

【００１９】さらにＪａｖａアーキテクチャは、Ｊａｖａ仮想マシンというインタプリタ機構によりプログラムを解釈実行する、という仕組みを持ち、機器のハードウェアやＯＳに依存しないプログラムを開発できるため、家電分野においてプログラム配信を実現するのに有望なアーキテクチャとみなされている。

【００２０】Ｊａｖａは、ネットワークを用いたプログラム（コンテンツ）配信の際に、セキュリティに関して高い安全性を保持していることでも知られている。具体的にはＪａｖａは、以下のような検証機構を持っている。

【００２１】（ＬａｕｒａＬｅｍａｙ，ｃｈａｒｌｅｓＬ．Ｐｅｒｋｉｎｓ，"ＴｅａｃｈＹｏｕｒｓｅｌｆＪａｖａｉｎ２１ｄａｙｓ" ＰｒｅｎｔｉｃｅＨａｌｌ，１９９６）１．Ｊａｖａ言語そのものが安全性の高い言

語仕様を持っており、Ｊａｖａ言語によって悪意ある変造コードを作成することは困難になっている。例えば、メモリに対する参照値などを、プログラマは直接操作することはできない。２．Ｊａｖａ仮想マシンは、実行形式（Ｊａｖａクラスファイル）が正しい規則に沿って記録されているかどうかを検証する機構を備える。これにより、変造された実行形式を実行解釈することを、Ｊａｖａ仮想マシンは拒絶する。３．仮想マシンのプログラム・ローダは記憶範囲をグループ別に管理しており、「保護の弱い」記憶範囲に記録されたクラスファイル（実行形式）が、「より保護の強い」記憶範囲に記録されたクラスファイルを侵害したり、許可なく参照したりすることはできない。例えば、インターネットから配信されたクラスファイルが、ローカルな領域に記録されていたクラスファイルに、許可なく置き換えられたりすることはない。４．ＪａｖａＡＰＩが提供する機能群は、プログラムが破壊的な動作を行なえないように設計されている。

【００２２】以上のようにＪａｖａは、開発言語からプログラム実行系までの各レベルにおいて、強固な検証機構を有しているが、このような機能を持たせたため、Ｊａｖａ仮想マシンはその動作のために多大な使用メモリ量と高性能のプロセッサを必要としている。

【００２３】例えば２を実現するために、仮想マシンは実行形式中のＪａｖａの命令コードに対してフロー解析（Ａ．Ａｈｏ，Ｒ．Ｓｅｔｈｉ，Ｊ．Ｕｌｌｍａｎ，"ｃｏｍｐｉｌｅｒＰｒｉｎｃｉｐｌｅｓ，Ｔｅｃｈｉｎｉｑｕｅｓ，ａｎｄＴｏｏｌｓ"，Ａｄｄｉｓｏｎ－Ｗｓｅｌｅｙ，１９８６）を行わなければならず、これはＪａｖａ仮想マシンの実装コードサイズと使用メモリ量の増大を招く（試算によると、２．の検証機構を備えない場合の総使用メモリ量と比較して、１．５～３．０倍増大する）。

【００２４】さらにこれらの検証処理は、プログラムの実行中にもしばしば発生するものであるため、Ｊａｖａ仮想マシンの処理の負荷を大きくし、高速にプログラムを動作させることへの妨げにもなる。

【００２５】家電機器分野ではコストに対する要求は非常に厳しく、コンピューティング分野以上に省資源・低コスト・低消費電力が強く要求されるものが大多数である。

【００２６】例えば家電機器にＪａｖａを搭載させるアプローチの代表的なものに、サンマイクロシステムズが家電組み込み向けに作成した仕様であるＥｍｂｅｄｄｅｄＪａｖａアーキテクチャがあるが、プログラムの動作のためには最低、ＲＡＭ５１２Ｋバイト、ＲＯＭ５１２Ｋバイト、２５ＭＨｚ以上のＣＰＵを必要としている（ＤｅｅｐａｋＭｕｌｃｈａｎｄａｎｉ，"ＪａｖａｆｏｒＥｍｂｅｄｄｅｄＳｙｓｔｅｍｓ"，ＩＥＥＥＩＮＴＥＲＮＥＴＣＯＭＰＵＴＩＮＧ，Ｍａｙ－ｊｕｎｅ１９９８）。これはハイエンドの情報家電機器に対して搭載可能、というレベルを実現したに留まるものであり、必要メモリ量は依然として一般の家電機器にとっては大きなサイズである。

【００２７】特に、携帯電話や白物家電などには、低動作周波数の８／１６ｂｉｔマイコンや、数十ＫＢ程度のメモリしか搭載しないものが主流であり、このような機器に対してはＥｍｂｅｄｄｅｄＪａｖａを適用することは不可能である。

【００２８】以上に示される通り、Ｊａｖａに代表されるような、検証機構を実行環境に搭載してネットワークを流れるプログラムの安全性を保証するというアプローチは、一般の家電機器全般に適用するには非常に困難である。

【００２９】本発明は、プログラム配信のためのネットワーク機能を備えた家電機器における問題である、ダウンロードしたプログラムの安全性・秘匿性・完全性の保証を、可能な限り低コストで簡便に実現するための手段を提供するものである。

【００３０】

【課題を解決するための手段】本発明の装置は、外部伝送路を通して、プログラムと、プログラムの発信元の識別情報を受信する。装置は受信した識別情報を判定し、その結果により、プログラムが安全なものであるか、それとも安全性の保障できない、実行環境とユーザに危険を及ぼす可能性のあるものであるか、を判断する。

【００３１】安全と判断したプログラムに対しては、解釈実行を許可する。この解釈実行の手段はプログラムに対して特別に検証機構による処理を必要としないものを想定している。

【００３２】よってプログラムを、使用メモリ量を少なく抑えたまま、高速に動作させることができる。

【００３３】安全性が保証できないプログラムに対しては、本発明は以下のような対処方法を提供している。

【００３４】１）安全性が保証できないプログラムは一切解釈実行させない。

【００３５】この方法を取る場合、プログラム実行装置は危険なプログラムを一切動作させないため、検証機構を備える必要はない。プログラムの解釈実行のために必要な資源量は小さく抑えたまま、安全と判断されたプログラムは全て高速に動作させることができる。

【００３６】２）ユーザによって与えられた、プログラムの解釈実行を許可する、あるいは禁止する、のいずれかを示す情報を元に、プログラムの解釈実行を決定する。

【００３７】この方法を取る場合、プログラム実行装置は安全性が保証できないプログラムを動作させるかどうかをユーザの判断に任せる。

【００３８】本来家電機器はその特徴上、信頼できる特定の（おそらくは機器を製造したメーカーの保証を受け

た）発信元からのみ、プログラムをダウンロードし実行することを推奨するものである。そのような背景においても、ユーザが開かれたネットワークから、プログラム（コンテンツ）のダウンロードを行うというのは、例外的な状況下においてであると考えられる。

【００３９】つまり、ユーザにとって何らかの理由によって有益と思われるプログラム・コンテンツを、その機器を使用して獲得したい場合である。

【００４０】開かれたネットワークから、プログラム（コンテンツ）を獲得する場合、ユーザはそのダウンロード対象プログラム・コンテンツが安全なものであるとの知識を持っていることが前提であり、ユーザは自己の責任においてその行為を行わなければならない。

【００４１】２）の方法は、そのユーザの判断を、プログラム実行装置に反映させる方式である。

【００４２】プログラムはユーザの自己責任によってプログラムを解釈実行させるため、検証処理は行われないことが前提となり、プログラムの解釈実行のために必要な資源量は小さく抑えたまま、プログラムを高速に動作させることができる。

【００４３】３）ダウンロードしたプログラムが危険なものである可能性があることをユーザに知らせた後、２）を行う。

【００４４】２）と同じであるがプログラムを動作させる前に、判断を促すための通告をユーザに対して行うことができる。

【００４５】ユーザはそれを用い、より正確な判断で、プログラムの解釈実行の許可や禁止の情報を装置に対して与えることができる。

【００４６】４）プログラム実行装置に、プログラムの解釈実行のための、従来とは別の第２の手段を用意し、安全性が保証できないプログラムはその第２の手段により実行することにする。

【００４７】安全性が保障されるプログラムとされないプログラムの解釈実行のための手段を分離する方法である。

【００４８】この場合、安全性が保証されないプログラムを解釈実行する手段は、検証処理を動作させつつプログラムを解釈実行すること、あるいは、プログラム実行装置の他の手段には影響を与えず独立して存在していることが望ましい。

【００４９】これにより、安全なプログラムを高速に動作させることを保証したまま、危険なプログラムもまた動作させることができる。５）２）と４）の方式を同時にとるもの６）３）と４）の方式を同時にとるもの本発明はこれらの手段を、プログラムを実行する機器に対して提供するものである。

【００５０】

【発明の実施の形態】本発明は、外部伝送路に接続されたプログラム実行環境全般に適用可能なものであるが、特に省資源・低コストが要求される家電機器への適用を強く意図したものである。

【００５１】ここでは特に、ディスプレイを装備したインターネットに接続可能な携帯端末に対して、本発明を適用した例を適宜加えながら、説明を行っていく。

【００５２】以下、本発明の実施の形態について、図面を用いて説明する。

【００５３】まず図１は、請求項１に記載のプログラム実行装置の構成を示したものである。

【００５４】図１の装置の処理のフローチャートを図５に示す。

【００５５】これらについて順に説明していく。

【００５６】外部伝送路１０１から受信する、プログラム１０２と識別情報１０３は、例えばインターネットなどのネットワークを通じてダウンロードする。

【００５７】プログラムは、装置が備える実行系により解釈実行が可能な形式を持ったデータとして獲得することができる。例えば装置がＪａｖａ仮想マシンを備えているならば、その装置はＪａｖａの実行形式であるＪａｖａクラスファイルが実行可能であるため、Ｊａｖａクラスファイルを外部伝送路を通じてダウンロードする。

【００５８】識別情報とは、プログラムをダウンロードする時に、例えば、その発信元が特定されるものであるとする。本実施例では、説明を簡単にするために、これをプログラムの名前で代用する。もちろん他の情報であってもよい。

【００５９】識別情報として用いることが可能なものに、例えば、インターネットに接続した機器に固有に割り振られるＩＰアドレスや、Ｗｅｂ，ＦＴＰ，Ｇｏｐｈｅｒ，Ｎｅｗｓ，ＴＥＬＮＥＴなどの情報を指定する世界共通で一意なアドレスであるＵＲＬ（Ｕｎｉｆｏｒｍ ＲｅｓｏｕｒｃｅＬｏｃａｔｏｒ）がある。

【００６０】米国ＮｅｔｓｃａｐｅＣｏｍｍｕｎｉｃａｔｉｏｎｓＣｏｒｐｏｒａｔｉｏｎによるＳＳＬ（ＳｅｃｕｒｅＳｏｃｋｅｔｓＬａｙｅｒ）プロトコルでは、相手の認証や使用する暗号やデジタル署名のアルゴリズムなどに関するネゴシエーションを行ない，相互に認証してから，データの送受信を行っているが、もちろんＳＳＬに使用されるものに代表される暗号化された署名を識別情報とすることも可能である。もちろん、識別情報は上記に挙げたもの以外のものであってもよい。

【００６１】図５において最初の処理であるＰ１０５では、識別情報１０３を解読し、識別情報が、信頼されたプログラム発信元を示しているか否かを判定する。

【００６２】これは図１における、プログラム発信先判定手段１０５によって行われる処理である。

【００６３】この処理Ｐ１０５の具現は、何らかの形で識別情報を解釈して、その信頼性を判定できる必要がある。例えば以下のような形で、これを実現することができる。

【００６４】プログラム発信先判定手段１０５は、信頼できる発信元を示した全ての識別情報を記録したデータを保持している。信頼できるか否かの判定は、ダウンロードしてきた識別情報をこのデータと照合し、識別情報がデータに登録されているなら、この識別情報は信頼できると判定し、登録されていないなら信頼できないと判定する。

【００６５】識別情報として、ＳＳＬプロトコルで用いられるような暗号化された署名を用いている場合には、プログラム発信先判定手段１０５は、暗号の解読に成功するか否かによって、識別情報を信頼できるかできないかを決定することもできる。もちろん、信頼性の判定は他の手段で行うこともできる。

【００６６】処理Ｐ１０４は、プログラム１０２を解釈実行する。

【００６７】これは図１における、プログラム解釈実行手段１０４によって行われる処理である。

【００６８】本発明の備えるプログラム解釈実行手段１０４は、プログラムを解釈実行することが可能な環境が備える、従来の実行系と同じ仕組みであり、従来と同様にプログラムの実行解釈を行う。

【００６９】処理Ｐ１０６は、プログラム１０２がプログラム解釈実行手段１０４によって解釈実行されるのを禁止する。

【００７０】これは図１における、プログラム解釈実行禁止手段１０６によって行われる処理である。

【００７１】プログラム解釈実行禁止手段１０６の実現には、例えば以下のようなものが考えられる。

【００７２】プログラム解釈実行禁止手段１０６は内部に、図９に示されるテーブルを保持する。

【００７３】このテーブルのエントリは、ダウンロードしたプログラムの識別子と、それに対応する、プログラムの状態を示す値とで構成される。プログラムの状態を示す値は、ｐｅｒｍｉｔｅｄもしくはｆｏｒｂｉｄｅｎという２値のどちらかを取る。テーブルに記録される時の初期状態はｐｅｒｍｉｔｔｅｄであるとする。

【００７４】プログラム解釈実行手段１０４は、解釈実行を始める前に図９のテーブルを参照して、解釈実行しようとするプログラムの識別子を検索する。該当するプログラムの識別子のエントリの、プログラムの状態を示す値がｆｏｒｂｉｄｄｅｎであるか、あるいはエントリがテーブルにない場合には、そのプログラムの解釈実行を開始しないことにする。

【００７５】この場合、処理Ｐ１０６は、該当するプログラムの識別子のエントリの、プログラムの状態を示す値をｆｏｒｂｉｄｄｅｎに変更することによって実現できる。

【００７６】ここではプログラム解釈実行禁止手段１０６の実現にテーブルを用いて実施例を示したが、もちろんテーブル以外の構造、例えばリスト、木、ハッシュを用いて実現することも可能である。

【００７７】処理Ｐ１０４、あるいは処理Ｐ１０６が終了すると、図１で行われる処理全体が完了する。

【００７８】図２は、請求項５記載のプログラム実行装置の構成を示したものである。

【００７９】図２から、プログラム解釈実行許可確認手段１０７を除いた図面が、請求項２記載のプログラム実行装置の構成を示すものになる。

【００８０】図２の装置の処理のフローチャートを図６に示す。

【００８１】図６において、Ｐ２０７の処理を「何もしない」に置き換えたものが、請求項２記載の装置の処理のフローチャートになる。

【００８２】これらについて順に説明していく。

【００８３】外部伝送路１０１から受信する、プログラム１０２および識別情報１０３は、先に説明したものと同じである。

【００８４】図６において最初の処理であるＰ２０５は、図２におけるプログラム発信先判定手段１０５によって行われる処理であり、先に説明したＰ１０５と同じ処理を行う。

【００８５】処理Ｐ２０７は、プログラムの実行を許可するか否かをユーザに問い合わせる。

【００８６】これは図２における、プログラム解釈実行許可確認手段１０７によって行われる処理である。

【００８７】例えばこれは、装置が備えるディスプレイに、図１０のような表示を行うことで実現できる。

【００８８】処理Ｐ２０９は、プログラムの実行を許可するか否かを示す情報である、プログラム解釈実行許可情報１０８を得る。

【００８９】これは図２における、プログラム解釈実行許可入力手段１０９によって行われる処理である。

【００９０】装置のユーザは、処理Ｐ２０９が行われる前に、何らかの意思表明を装置に対して行う。プログラム解釈実行許可入力手段は、その意思表明を受け、装置内部のデータとして、プログラム解釈実行許可情報１０８を生成する。

【００９１】ユーザによる意思表明は、装置が備える入力装置、例えばボタン、スイッチ、ダイヤル、マウス、トラックボール、ジョイスティック、等による入力操作の組み合わせによって、あるいはこれらの操作を何もしないことによってなされる。

【００９２】例えば・ユーザは、”機能”ボタンを押した後、”０”ボタンを押す、という行為を、装置に対して行う。これによって装置は、全てのプログラムの実行が許可されたという情報を得る。・ユーザは、ダイヤルによって”ｍａｚｅ．ｃｊ”という文字列をディスプレイ上で選択した後に、”１”ボタンを押す。これによって装置は、”ｍａｚｅ．ｃｊ”という識別名を持つプログラムの実行が禁止されたという情報を得る。・ユーザ

は、処理Ｐ２０９が行われる前に、プログラムの実行の許可／禁止を指示する特定の入力を、何ら装置に対して与えなかったとする。これによって装置は、”ｍａｚｅ．ｃｊ”という識別名を持つプログラムの実行が禁止されたという情報を得る。

【００９３】ユーザの意思表明は、処理Ｐ２０７と連携して実施されても良い。例えば、・図１０の表示が行なわれた後に、”０”ボタンを押す、という行為を、装置に対して行う。これによって装置は、”ｍａｚｅ．ｃｊ”という識別名を持つプログラムの実行が許可されたという情報を得る。

【００９４】ここでは”０”，”１”，”機能”というボタンとダイヤル操作の組み合わせによる実施例を示したが、もちろんボタンの種類や押す順番は上に挙げたもの以外であっても良いし、入力装置の組み合わせも他のものであっても良い。

【００９５】処理Ｐ２０６Ａは、プログラム解釈実行許可情報１０８を解釈して、ユーザがプログラム１０２の実行を許可したか否かを判定する。

【００９６】これは図２において、プログラム解釈実行禁止手段１０６によって行われる処理である。

【００９７】処理Ｐ２０４は、プログラム１０２を解釈実行する。

【００９８】これは図２における、プログラム解釈実行手段１０４によって行われる処理であり、先に説明した処理Ｐ１０４と同じものである。

【００９９】処理Ｐ２０６Ｂは、プログラム１０２の解釈実行を禁止する。

【０１００】この処理は、図２のプログラム解釈実行禁止手段１０６によって行われる処理であり、先に説明した処理Ｐ１０６と同じものである。

【０１０１】処理Ｐ２０４、あるいは処理Ｐ２０６Ｂが終了すると、図２で行われる処理全体が完了する。

【０１０２】図３は、請求項３に記載のプログラム実行装置の構成を示したものである。

【０１０３】図３の装置の処理のフローチャートを図７に示す。

【０１０４】これらについて順に説明していく。

【０１０５】外部伝送路１０１から受信する、プログラム１０２および識別情報１０３は、先に説明したものと同じである。

【０１０６】図７において最初の処理であるＰ３０５は、図３におけるプログラム発信先判定手段１０５によって行われる処理であり、先に説明したＰ１０５と同じ処理を行う。

【０１０７】処理Ｐ３０４は、プログラム１０２をプログラム解釈実行手段１０４によって実行する。

【０１０８】これは図３における、プログラム解釈実行選択手段１１１と、プログラム解釈実行手段１０４によって行われる処理である。

【０１０９】処理Ｐ３１０は、プログラム１０２をプログラム解釈実行副手段１１０によって実行する。

【０１１０】これは図３における、プログラム解釈実行選択手段１１１と、プログラム解釈実行副手段１１０によって行われる処理である。

【０１１１】プログラム解釈実行副手段１１０は、プログラムを解釈実行することが可能な環境が備える、従来の実行系と同じ仕組みであり、従来と同様にプログラムの実行解釈を行うが、これはプログラム解釈実行手段１０４とは別の手段として装置内部に実現される。

【０１１２】プログラム解釈実行選択手段１１１の実現には、例えば以下のようなものが考えられる。

【０１１３】プログラム解釈実行選択手段１１１は内部に、図１１に示されるテーブルを保持する。

【０１１４】このテーブルのエントリは、ダウンロードしたプログラムの識別子と、それに対応する、プログラムの状態を示す値とで構成される。

【０１１５】プログラムの状態を示す値は、ｓｕｃｕｒｅｄもしくはｕｎｓｅｃｕｒｅｄという２値のどちらかを取る。

【０１１６】プログラムの実行の前に、プログラム解釈実行選択手段１１１は、プログラム発信先判定手段１０５の判定結果によって、信頼された発信元からのプログラムであればｓｅｃｕｒｅｄを、そうでなければｕｎｓｅｃｕｒｅｄを記録する。

【０１１７】プログラム解釈実行手段１０４は、解釈実行を始める前に図１１のテーブルを参照して、解釈実行しようとするプログラムの識別子を検索する。該当するプログラムの識別子のエントリの、プログラムの状態を示す値がｓｅｃｕｒｅｄである場合にのみ、そのプログラムの解釈実行を開始する。

【０１１８】プログラムの状態を示す値がｕｎｓｅｃｕｒｅｄであるか、あるいはエントリがテーブルにない場合には、そのプログラムの解釈実行はプログラム解釈実行副手段１１０によって実行される。

【０１１９】ここではプログラム解釈実行選択手段１１１の実現にテーブルを用いて実施例を示したが、もちろんテーブル以外の構造、例えばリスト、木、ハッシュを用いて実現することも可能である。

【０１２０】処理Ｐ３０４、あるいは処理Ｐ３１０が終了すると、図３で行われる処理全体が完了する。

【０１２１】図４は、請求項５記載のプログラム実行装置の構成を示したものである。

【０１２２】図４から、プログラム解釈実行許可確認手段１０７を除いた図面が、請求項４記載のプログラム実行装置の構成を示すものになる。

【０１２３】図４の装置の処理のフローチャートを図８に示す。

【０１２４】図８において、Ｐ４０７の処理を「何もしない」に置き換えたものが、請求項４記載の装置の処理

のフローチャートになる。
【０１２５】これらについて順に説明していく。
【０１２６】外部伝送路１０１から受信する、プログラム１０２および識別情報１０３の説明は、先に説明したものと同じである。
【０１２７】図８において最初の処理であるＰ４０５は、図４におけるプログラム発信先判定手段１０５によって行われる処理であり、先に説明したＰ１０５と同じ処理を行う。
【０１２８】処理Ｐ４０７は、プログラムの実行を許可するか否かをユーザに問い合わせる。
【０１２９】これは図４における、プログラム解釈実行許可確認手段１０７によって行われる処理であり、先に説明したＰ２０７と同じ処理を行う。
【０１３０】処理Ｐ４０９は、プログラムの実行を許可するか否かを示す情報である、プログラム解釈実行許可情報１０８を得る。
【０１３１】これは図４における、プログラム解釈実行許可入力手段１０９によって行われる処理であり、先に説明したＰ２０９と同じ処理を行う。
【０１３２】処理Ｐ４０６Ａは、プログラム解釈実行許可情報１０８を解釈して、ユーザがプログラム１０２の実行を許可したか否かを判定する。
【０１３３】これは図４において、プログラム解釈実行禁止手段１０６によって行われる処理である。
【０１３４】処理Ｐ４０４は、プログラム１０２をプログラム解釈実行手段１０４によって実行する。
【０１３５】これは図４における、プログラム解釈実行禁止選択手段１１２と、プログラム解釈実行手段１０４によって行われる処理である。
【０１３６】処理Ｐ４１０は、プログラム１０２をプログラム解釈実行副手段１１０によって実行する。
【０１３７】これは図４における、プログラム解釈実行禁止選択手段１１２と、プログラム解釈実行副手段１１０によって行われる処理である。
【０１３８】処理Ｐ４０６Ｂは、プログラム１０２の解釈実行を禁止する。
【０１３９】この処理は、図４のプログラム解釈実行禁止選択手段１１２によって行われる処理である。
【０１４０】プログラム解釈実行禁止選択手段１１２の実現には、例えば以下のようなものが考えられる。
【０１４１】プログラム解釈実行禁止選択手段１１２は内部に、図１２に示されるテーブルを保持する。このテーブルのエントリは、ダウンロードしたプログラムの識別子と、それに対応する、プログラムの状態を示す値とで構成される。
【０１４２】プログラムの状態を示す値は、ｓｕｃｕｒｅｄもしくはｐｅｒｍｉｔｔｅｄもしくはｆｏｒｂｉｄｄｅｎという３つの値のいずれかを取る。ここではエントリに最初に記録されるときの初期値はｓｅｃｕｒｅｄであるとする。
【０１４３】処理Ｐ４０６Ａにおいて、プログラム解釈実行選択手段１１１は、プログラム解釈実行許可情報１０８を解釈して、ユーザが実行を許可していれば、プログラムの状態を示す値にｐｅｒｍｉｔｔｅｄを、そうでなければｆｏｒｂｉｄｄｅｎを記録する。
【０１４４】プログラム解釈実行手段１０４は、プログラム１０２の解釈実行を始める前に図１１のテーブルを参照して、解釈実行しようとするプログラムの識別子を検索する。該当するプログラムの識別子のエントリの、プログラムの状態を示す値がｓｅｃｕｒｅｄである場合にのみ、そのプログラムの解釈実行を開始する。
【０１４５】プログラムの状態を示す値がｐｅｒｍｉｔｔｅｄである場合には、プログラム１０２はプログラム解釈実行副手段１１０によって解釈実行される。
【０１４６】プログラムの状態を示す値がｆｏｒｂｉｄｄｅｎであるか、あるいはエントリがテーブルにない場合には、プログラム１０２は解釈実行されない。
【０１４７】ここではプログラム解釈実行禁止選択手段１１２の実現にテーブルを用いて実施例を示したが、もちろんテーブル以外の構造、例えばリスト、木、ハッシュを用いて実現することも可能である。
【０１４８】処理Ｐ４０４、あるいは処理Ｐ４１０、あるいは処理Ｐ４０６Ｂが終了すると、図４で行われる処理全体が完了する。
【０１４９】以上が本発明の実施の形態である。
【０１５０】
【発明の効果】以上のように本発明は、外部伝送路を通して、プログラムと、プログラムの発信元の識別情報を受信し、その識別情報が、信頼できる発信元を示しているかどうかを判定する。
【０１５１】判定の結果、信頼できる発信元のものであることが判別できない場合、実行環境は、そのことをユーザ（実行環境の使用者）に知らせ、ユーザによって与えられた情報を元に、プログラムの解釈実行を決定する。もしくは、従来の解釈実行手段とは異なる、第２の解釈実行手段によって、そのプログラムを解釈実行する。
【０１５２】このような機構を設けることにより、プログラムを実行する装置は、安全であると保証されるプログラムは従来通り、使用資源が少なくかつ高速な解釈実行を行うことができる。かつダウンロードしたプログラムの安全性・秘匿性・完全性の保証を、装置の低コスト・省資源性を保ったまま行うことができる。
【図面の簡単な説明】
【図１】本発明のプログラム実行装置の構成例を示す図
【図２】本発明のプログラム実行装置の構成例を示す図
【図３】本発明のプログラム実行装置の構成例を示す図
【図４】本発明のプログラム実行装置の構成例を示す図
【図５】本発明のプログラム実行装置の動作例を示すフ

15　　16

ローチャート

【図６】本発明のプログラム実行装置の動作例を示すフローチャート

【図７】本発明のプログラム実行装置の動作例を示すフローチャート

【図８】本発明のプログラム実行装置の動作例を示すフローチャート

【図９】プログラム解釈実行禁止手段の構成例を示す図

【図１０】プログラム解釈実行許可確認手段による表示例を示す図

【図１１】プログラム解釈実行選択手段の構成例を示す＊図

【図１２】プログラム解釈実行禁止選択手段の構成例を示す図

【符号の説明】

１０１　通信伝送路

１０２　プログラム

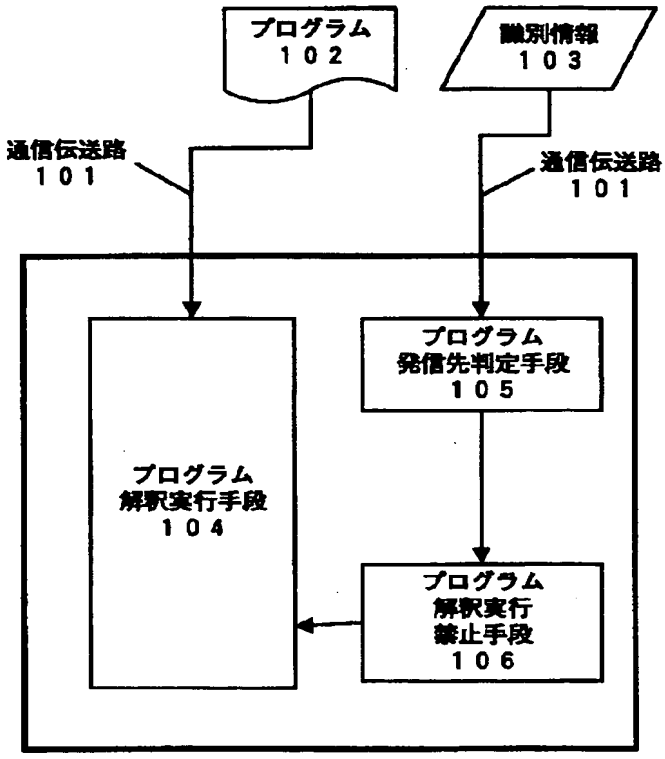
１０３　識別情報

１０４　プログラム解釈実行手段

１０５　プログラム発信先判定手段

１０６　プログラム解釈実行禁止手段

【図１】

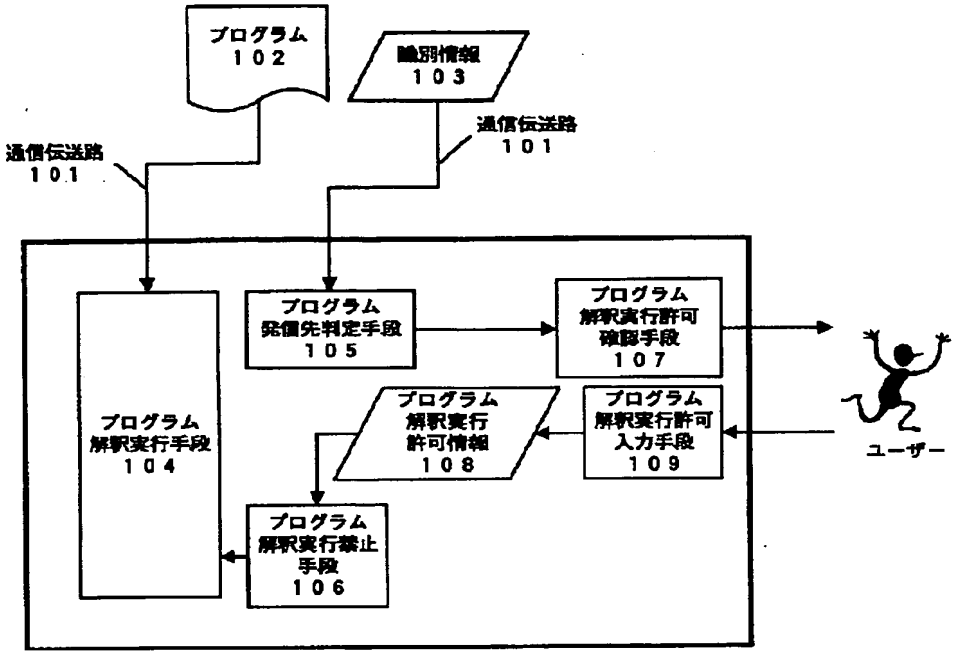


プログラム実行装置・その１

【図１０】

プログラム解釈実行許可確認手段の実施例

【図２】



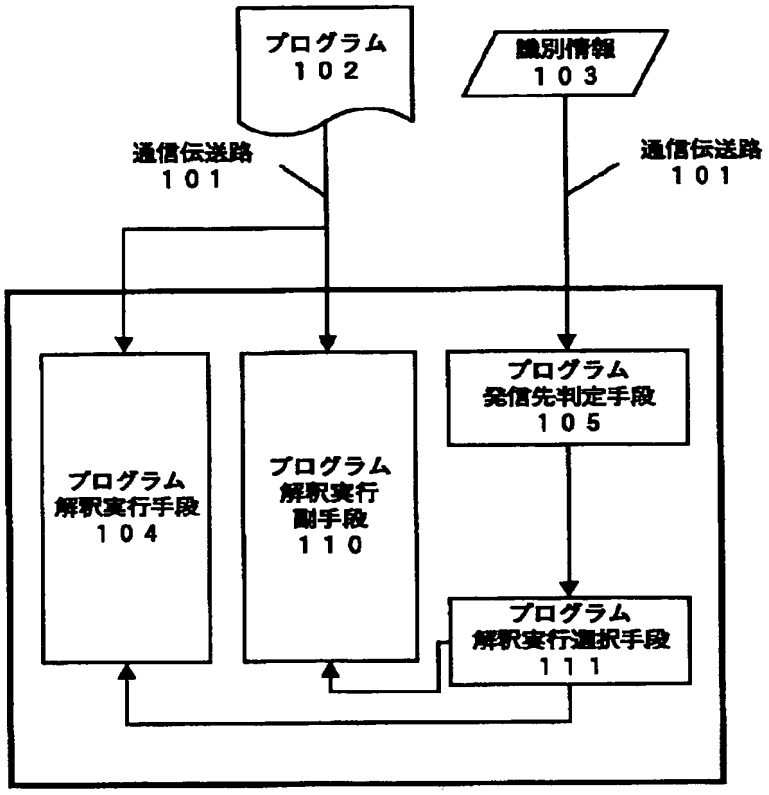
プログラム実行装置・その２

【図９】

プログラムの識別子	プログラムの状態
ButtonCrusher.cj	forbidden
Maze.cj	permitted
ClipBoard.cj	permitted
AddressWriter.oj	forbidden
DigitalClock.cj	permitted
⋮	⋮

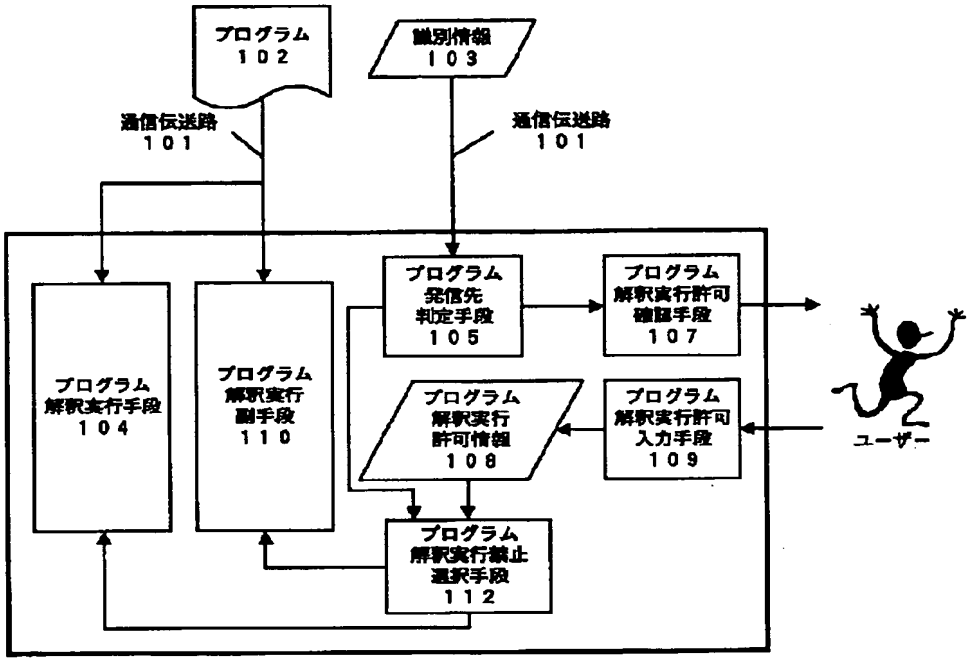
プログラム解釈実行禁止手段の実施例

【図３】



プログラム実行装置・その３

【図４】



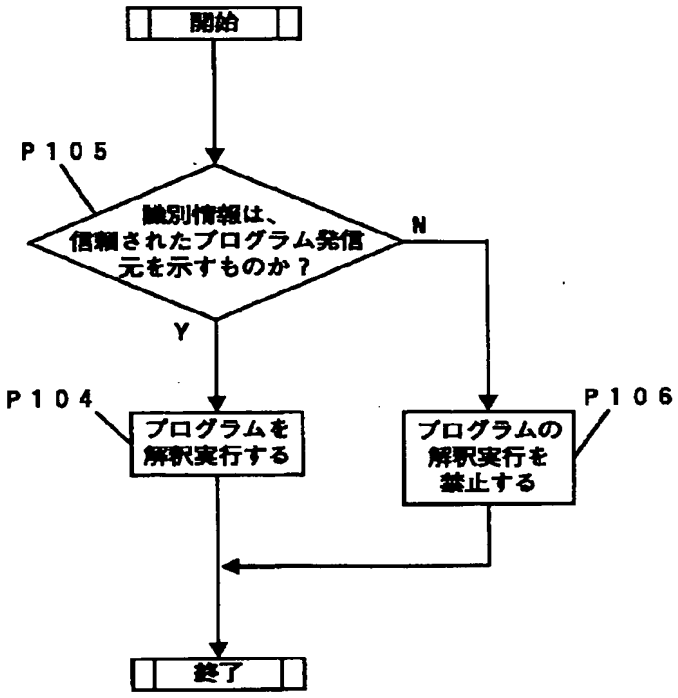
プログラム実行装置・その４

【図１１】

プログラムの識別子	プログラムの状態
ButtonCrusher.cj	secured
Maze.cj	unsecured
ClipBoard.cj	unsecured
AddressWriter.cj	secured
DigitalClock.cj	unsecured
⋮	⋮

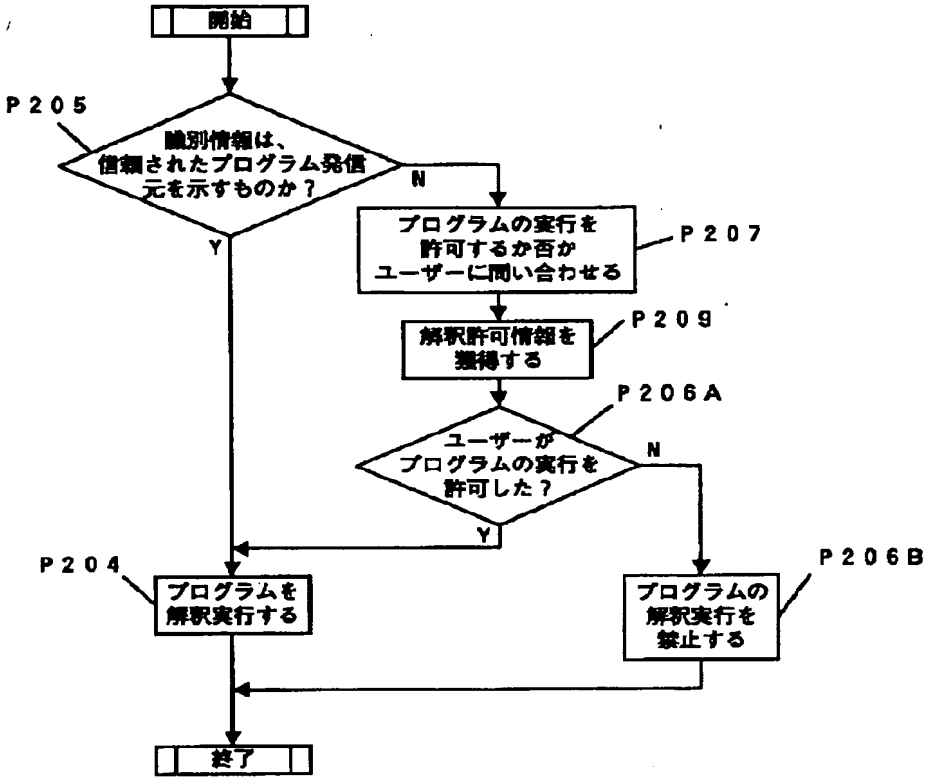
プログラム解釈実行選択手段の実施例

【図５】



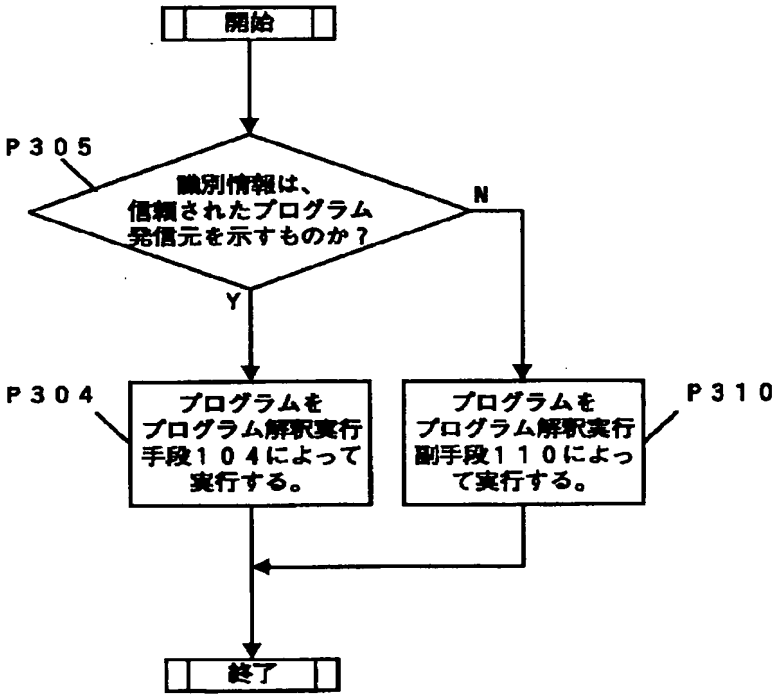
プログラム実行方式の処理手順・その１

【図６】



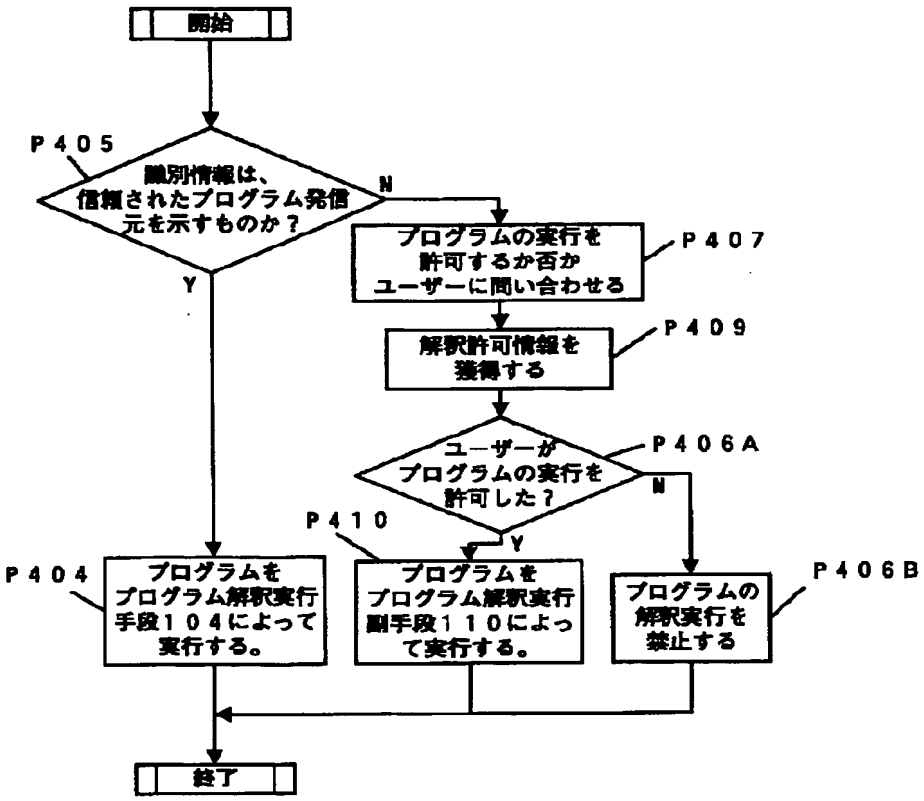
プログラム実行方式の処理手順・その２

【図７】



プログラム実行方式の処理手順・その３

【図８】



プログラム実行方式の処理手順・その４

【図１２】

プログラムの識別子	プログラムの状態
ButtonCrusher.cj	forbidden
Maze.cj	secured
ClipBoard.cj	permitted
AddressWriter.cj	forbidden
DigitalClock.cj	secured
⋮	⋮

プログラム解釈実行禁止選択手段の実施例

(11) Japanese Unexamined Patent Application Publication No. 2001-117769
(43) Publication Date: April 27, 2001
(21) Application No. 11-297759
(22) Application Date: October 20, 1999
(71) Applicant: Matsushita Electric Industrial Co., Ltd.
(72) Inventor: Tomokazu KANAMARU
(74) Agent: Patent Attorney, Fumio IWAHASHI et al.

(54) [Title of the Invention] APPARATUS FOR INTERPRETING PROGRAM

(57) [Abstract]

[Object] To realize the assurance of security, confidentiality, and integrity of downloaded programs simply and at low cost in household electrical appliances having the function of connecting to networks to receive distributed programs.

[Solving Means] An apparatus for executing a program includes interpreting means 104, secondary interpreting means 110, source determining means 105, input means 109, and program interpretation forbiddance selecting means 112. If the source determining means 105 determines that identifying information of a program 102 indicates a trusted source, the program interpretation forbiddance selecting means 112 permits the interpreting means 104 to interpret the program 102, and if not, the program interpretation forbiddance selecting means 112 forbids the interpreting means 104 from interpreting the program 102.

[Claims]

[Claim 1] An apparatus for executing a program, the apparatus receiving a program transmitted through an external communication channel and identifying information transmitted through the external communication channel, the apparatus comprising:

interpreting means for interpreting the program;

source determining means for determining whether the identifying information indicates that the program is transmitted from a trusted source; and

program-interpretation forbidding means,

wherein the program-interpretation forbidding means forbids the interpreting means from interpreting the program in accordance with the determination performed by the source determining means.

[Claim 2] An apparatus for executing a program, the apparatus receiving a program transmitted through an external communication channel and identifying information transmitted through the external communication channel, the apparatus comprising:

interpreting means for interpreting the program;

source determining means for determining whether the identifying information indicates that the program is transmitted from a trusted source;

program-interpretation forbidding means; and

input means by which a user inputs permission information indicating either that the program is permitted to be interpreted or that the program is forbidden from being interpreted,

wherein the program-interpretation forbidding means forbids the interpreting means from interpreting the program in accordance with the permission information.

[Claim 3] An apparatus for executing a program, the apparatus receiving a program transmitted through an external communication channel and identifying information transmitted through the external communication channel, the apparatus comprising:

interpreting means for interpreting the program;

secondary interpreting means being second means for interpreting a program;

source determining means for determining whether the identifying information indicates that the program is transmitted from a trusted source; and

program-interpretation selecting means,

wherein the program-interpretation selecting means selects permitting or forbidding the program from being executed by the interpreting means in accordance with the determination performed by the source determining means, and in the case that the program is forbidden from being interpreted by the interpreting means, the program is executed by the secondary interpreting means.

[Claim 4] An apparatus for executing a program, the apparatus receiving a program transmitted through an external communication channel and identifying information transmitted through the external communication channel, the apparatus comprising:

interpreting means for interpreting the program;

secondary interpreting means being second means for interpreting a program;

source determining means for determining whether the identifying information indicates that the program is transmitted from a trusted source; and

input means by which a user inputs permission information indicating either that the

program is permitted to be interpreted or that the program is forbidden from being interpreted; and

program-interpretation forbiddance selecting means,

wherein, in the case that the determination performed by the source determining means is that the program is transmitted from a trusted source, the program-interpretation forbiddance selecting means permits the interpreting means to interpret the program, and, in the case that the determination performed by the source determining means is that the program is not transmitted from a trusted source, the program-interpretation forbiddance selecting means forbids the interpreting means from interpreting the program,

wherein, in the case that the determination performed by the source determining means is that the program is not transmitted from a trusted source, the program-interpretation forbiddance selecting means selects permitting or forbidding the secondary interpreting means from executing the program in accordance with the permission information.

[Claim 5] The apparatus according to claim 2 and claim 4, further comprising inquiring means for inquiring of a user whether the program is permitted to be interpreted in the case that the determination performed by the source determining means is that the program is not transmitted from a trusted source.

[Claim 6] The apparatus according to claims 1, 2, 3, 4, and 5, wherein the identifying information includes an IP address, and the source determining means performs the determination using the IP address.

[Claim 7] The apparatus according to claims 1, 2, 3, 4, and 5, wherein the identifying information includes a uniform resource locator (URL), and the source determining means 105 performs the determination using the URL.

[Claim 8] The apparatus according to claims 1, 2, 3, 4, and 5, wherein the identifying information includes an encrypted signature, and the source determining means performs the determination using the encrypted signature.

[Detailed Description of the Invention]

[0001]

[Technical Field of the Invention] The present invention relates to an apparatus and method for verifying a downloaded program in an environment for downloading a program over network communications and for interpreting the program for use in particular applications, such as household electrical appliances.

[0002]

[Description of the Related Art] In recent years, the need for the ability to connect to networks to deploy programs distributed therethrough in household electrical appliances has grown.

[0003] The distribution of programs over networks facilitates updating and adding new functions to household electrical appliances and leads to new services, and therefore, it offers many advantages.

[0004] In the field of computing, such program distribution has been widely used.

[0005] A typical example is a network between a web server on the Internet and a personal computer (PC) including a browser. Anyone who uses this network can freely download content from sites on the network and use them. Networks of this type are referred to as "open networks" herein.

[0006] In open networks, users can freely download programs existing in the networks and execute them.

[0007] However, in open networks, verification of the following items relies on network users: 1. The source of connection is the one intended by a user (authentication and permission); 2. The content is never altered or intercepted in transmission paths (integrity and confidentiality); and 3. The content will never pose a threat to the environments in which it is used (security).

[0008] In other words, the items 1 to 3 described above are not entirely assured in open networks. Therefore, users may unknowingly download and execute dangerous programs created by a malicious third party.

[0009] In contrast to open networks, there are networks that assure the items 1 to 3 described above by requiring encryption of the content and/or user authentication in downloading the distributed content. Networks of this type are referred to as "closed networks" herein.

[0010] In general, household electrical appliances require high reliability and security. Therefore, in the distribution of programs used for household electrical appliances, it is desired that only a secure program sent from a trusted source be downloaded.

[0011] In addition, in the distribution of programs used for household electrical appliances, billing and data protection must be performed accurately and securely in many cases. Open networks are insufficient for fulfilling these requirements. Therefore, it is preferable that the distribution of content for household electrical appliances use closed networks.

[0012] Actually, current networks for use in household electrical appliances are "closed networks", which assure the items 1 to 3 by some means, in most cases.

[0013] For example, in various current services employing telephone networks, billing for network utilization through calls needs to be performed accurately. As a result, such services are implemented by closed networks. Closed networks may be further secured and become less vulnerable by subjecting the content to other processing, such as encryption.

[0014]

[Problems to be Solved by the Invention] However, even in the field of household electrical appliances, some devices are already well on their way to connecting to open network environments, typified by services of cellular telephones capable of connecting to the Internet. It is conceivable that household electrical appliances will have increased compatibility with open networks, typified by the Internet, in the future.

[0015] Under these circumstances, as described above, it is difficult to ensure that household electrical appliances download and execute a secure program only from a trusted source using closed networks.

[0016] One solution is that execution environments for interpreting programs have a mechanism for verifying that programs are not dangerous.

[0017] A typical architecture implementing this solution is Java.

[0018] The Java programming language has many advantages as a development language. Examples of such advantages include high efficiency in development as an object-oriented language, language specifications producing less bugs, portability, and abundant development environments.

[0019] In addition, the Java architecture has a mechanism in which Java virtual machines serving as interpreter mechanisms interpret programs. Thus, the Java architecture is capable of developing programs that are independent of hardware and operating systems in devices. Therefore, the Java architecture is considered to be promising for realizing the distribution of programs in the field of household electrical appliances.

[0020] It is also known that Java holds high security in distributing programs (content). Specifically, Java contains the verification mechanism described below.

[0021] (Laura Lemay, Charles L. Perkins, "Teach Yourself Java in 21 days" Prentice Hall, 1996) 1. The Java language itself has language specifications having high security, and therefore, creating malicious altered code by using the Java language is difficult. For example, programmers are unable to directly manipulate reference values in memory. 2. Java virtual machines include a mechanism for verifying that execute forms (Java class files) are written under correct rules. This prevents the Java virtual machines from interpreting programs with altered execute forms. 3. Program loaders in the Java virtual machines manage storage areas in groups. Class files (execute forms) in a "less protected" storage area are unable to invade those in a "highly protected" storage area and unable to refer to them without permission. For example, class files stored in a local area are not replaced with those deployed over the Internet without permission. 4. Functions provided by the Java API are designed to prevent programs from performing destructive operations.

[0022] As described above, Java includes less-vulnerable verification mechanisms in each level ranging from the development language to the program execution system. To perform these functions, however, the Java machines require large memory usage and a high performance processor.

[0023] For example, in order to realize item 2 above, the Java virtual machines must perform flow analysis on instruction codes in execute forms (A. Aho, R. Sethi, J. Ullman, "Compiler Principles, Techiniques, and Tools", Addison-Wseley, 1986). This leads to an increase in the size of code implemented and the amount of memory usage in the Java virtual machines. (According to a trial calculation, the increase is 1.5 times to 3 times the total amount of memory usage in the Java virtual machines not including the verification mechanism described in item 2.

[0024] In addition, the verification processing often occurs during program execution. This increases the processing load in the Java virtual machine and thus prevents programs from running at high speed.

[0025] In the field of household electrical appliances, a lower price is extremely desirable. Many appliances require high resource saving, low cost, and low power consumption, more so than computing devices.

[0026] For example, a typical approach to installing Java on household electrical appliances is an embedded Java architecture, invented by Sun Microsystems, for use in household electrical appliances. This needs at least 512K bytes of RAM, 512K bytes of ROM, and a 25-MHz CPU to run programs (Deepak Mulchandani, "Java for Embedded Systems", IEEE INTERNET COMPUTING, May-June, 1998). These specifications are applicable to some high-end appliances. However, the necessary amount of memory is still large in general household electrical appliances.

[0027] In particular, most cellular phones and white goods have an 8/16-bit microprocessor having a low operating frequency and memory of the order of several tens of kilobytes. It is impossible to apply Embedded Java to such devices.

[0028] As described above, the approach, typified by that in Java, to embedding the verification mechanism in execution environments to ensure the security of programs distributed on networks is extremely difficult in general household electrical appliances.

[0029] An object of the present invention is to provide means for realizing the assurance of security, confidentiality, and integrity of downloaded programs simply and at lower cost.

[0030]

[Means for Solving the Problems] According to the present invention, an apparatus receives, through an external communication channel, a program and identifying information indicating a source sending the program. In accordance with the received identifying information, the apparatus determines whether the program is secure or not so as to be possibly dangerous to the execution environment and user.

[0031] The apparatus permits interpreting a program that is determined to be secure. The means for interpreting not requiring particular processing by a verification mechanism with respect to the program is assumed.

[0032] Therefore, the apparatus can execute the program at high speed while suppressing the amount of memory used.

[0033] The present invention provides the following processing with respect to a program whose security is not assured.

[0034] (1) No insecure program is interpreted.

[0035] In this case, the apparatus for executing a program runs no insecure program, so that a verification mechanism is not required. Therefore, the apparatus can run every secure program at high speed while suppressing the amount of resources required for interpreting the program.

[0036] (2) The execution of a program is determined in accordance with information provided by a user, the information indicating permission or forbiddance of interpreting the program.

[0037] In this case, the apparatus leaves the user to select whether to execute an insecure program.

[0038] Essentially, in household electrical appliances, a program is recommended to be downloaded only from a trusted source (probably, authorized by the manufacturer that produced the household electrical appliance to which the apparatus is applied) and executed. In view of these circumstances, downloading a program (content) over an open network is an exceptional case.

[0039] In other words, in this case, the user needs to acquire a beneficial program (content) for some reason using the appliance.

[0040] In order to acquire a program (content) over an open network, knowledge about the security of the program (content) to be downloaded is assumed and the user must download it at his or her own risk.

[0041] In the processing (2), the selection of the user is reflected in the processing of the apparatus.

[0042] The program is interpreted by the user at his or her own risk, and therefore, no verification processing is a precondition. Therefore, the apparatus can run the program at high speed while suppressing the amount of resources required for interpreting the program.

[0043] (3) The apparatus informs the user of the possibility that a downloaded program may be dangerous, and then performs the processing (2).

[0044] This processing is similar to the processing (2), but, in this processing (3), the apparatus can provide a notification to the user for prompting the selection before executing the program.

[0045] The user employs information from the notification, so that the user performs the selection more accurately to provide information indicating permission or forbiddance of executing the program.

[0046] (4) The apparatus includes second interpreting means separated from the existing interpreting means. An insecure program is executed by the second means.

[0047] This processing has separate means, one for interpreting a secure program and the other for

an insecure program.

[0048] In this case, preferably, the second means for interpreting an insecure program interprets the insecure program while performing verifying processing or the second means exists independently so as to have no adverse effect on other means in the apparatus.

[0049] Therefore, the apparatus can run dangerous programs while assuring running of secure programs at high speed. (5) The processing is made up of the processing (2) and (4). (6) The processing is made up of the processing (3) and (4). The present invention provides these means in an apparatus for executing a program.

[0050]

[Embodiments] The present invention is applicable to general program-execution environments connected to external communication channels. In particular, the present invention is intended to be applied to household electrical appliances requiring higher resource saving and lower cost.

[0051] Embodiments in which the present invention is applied to a mobile terminal including a display and being capable of being connected to the Internet are explained.

[0052] The embodiments are described with reference to the drawings.

[0053] Fig. 1 shows the structure of an apparatus for executing a program according to claim 1.

[0054] Fig. 5 shows a flowchart of processing in the apparatus shown in Fig. 1.

[0055] The structure and processing are described step by step.

[0056] A program 102 and identifying information 103 are received through an external communication channel 101 by downloading them through, for example, networks, such as the Internet.

[0057] The program can be acquired as data having a form interpretable by an execution system included in the apparatus. For example, if the apparatus includes a Java virtual machine, the apparatus is able to execute Java class files, which are an executable form of Java, and thus, a Java class file is downloaded through the external communication channel.

[0058] The identifying information is used to identify a program by, for example, locating a source from where a program is sent when the program is downloaded. For the sake of simplicity, the name of the program is used as the identifying information in this embodiment. The identifying information may be other information.

[0059] Examples of such identifying information include IP addresses, one being a unique address assigned to each device on the Internet, and uniform resource locators (URLs), one being a worldwide unique address for identifying information in the hypertext transfer protocol (HTTP), file transfer protocol (FTP), Gopher, News, and Telnet.

[0060] According to the secure sockets layer (SSL) developed by Netscape Communications Corporation in the United States, cross-verification is completed through negotiations regarding algorithms for authentication, encryption, and digital signatures before data is exchanged. An encrypted signature, typified by the one used in the SSL, may be used as the identifying information. Of course, it will be understood that the identifying information may be another type of information other than the information described above.

[0061] In the first process P105 in Fig. 5, the identifying information 103 is read and it is determined whether the identifying information 103 indicates a trusted source for transmitting a program.

[0062] This process is performed by source determining means 105 shown in Fig. 1 for

determining whether a source sending a program is trusted.

[0063] In order to realize process P105, it is necessary to read the identifying information in some way and determine whether the source is trusted. This is achieved in the following way, for example.

[0064] The source determining means 105 retains data in which all identifying information indicating trusted sources is recorded. The determination is performed by checking identifying information for a downloaded program with this data and, if the identifying information is present in the data, determining that the identifying information indicates a trusted source, if not, determining that identifying information does not indicate a trusted source.

[0065] In the case that an encrypted digital signature, such as the one used in the SSL, is used as identifying information, the source determining means 105 can determine whether the identifying information indicates a trusted source by success or failure in decryption of the encrypted signature. The determination may be performed by other means.

[0066] In process P104, the program 102 is interpreted.

[0067] This process is performed by program interpreting means 104 shown in Fig. 1.

[0068] The interpreting means 104 included in the present invention has a mechanism similar to known execution systems included in environments capable of interpreting a program, and interprets a program in a manner similar to the known systems.

[0069] In process P106, the program 102 is forbidden from being interpreted by the interpreting means 104.

[0070] This process is performed by program interpretation forbidding means 106 shown in Fig. 1.

[0071] The program interpretation forbidding means 106 is realized by, for example, the following.

[0072] The program interpretation forbidding means 106 retains the table illustrated in Fig. 9 therein.

[0073] An entry in this table is made up of identification data of a downloaded program and a corresponding value indicating the status of the program. The value indicating the status of the program can take only two possible values: "permitted" or "forbidden". The initial value when the entry is recorded for the first time is set to "permitted".

[0074] The interpreting means 104 refers to the table shown in Fig. 9 to search for identification data for a program to be interpreted before starting interpreting the program. If the value indicating the program status associated with the identification data is "forbidden" or an entry of the identification data is not present in the table, the program is not interpreted.

[0075] In this case, process P106 is realized by changing the value indicating the status to "forbidden".

[0076] In this embodiment, the program interpretation forbidding means 106 uses the table, but it may use another structure, such as a tree, a list, or a hash.

[0077] When process P104 or P106 is finished, the entire processing shown in Fig. 1 is then completed.

[0078] Fig. 2 shows the structure of an apparatus for executing a program according to claim 5.

[0079] The block diagram, in which inquiring means 107 for inquiring of a user whether the program is permitted to be interpreted is removed from the structure shown in Fig. 2, shows an apparatus for executing a program according to claim 2.

[0080] Fig. 6 shows a flowchart of processing in the apparatus shown in Fig. 2.

[0081] The flowchart, in which process P207 is replaced with "performing nothing" in Fig. 6, shows processing in the apparatus according to claim 2.

[0082] The structure and processing are described step by step.

[0083] The program 102 and the identifying information 103 received through the external communication channel 101 are the same as those described above.

[0084] The first process P205 in Fig. 6 is performed by the source determining means 105 shown in Fig. 2 and is the same as process P105 described above.

[0085] In process P207, an inquiry as to whether to permit executing the program is made to a user.

[0086] This process is performed by the inquiring means 107 shown in Fig. 2.

[0087] This process is realized by, for example, displaying information shown in Fig. 10 on a display screen included in the apparatus.

[0088] In process P209, permission information 108 indicating whether executing a program is permitted or not is acquired.

[0089] This process is performed by input means 109 by which a user inputs permission information, shown in Fig. 2.

[0090] Before process P209, the user of the apparatus provides to the apparatus a response indicating the user's selection. Upon receipt of the response, the input means 109 creates the permission information 108 in the form of data within the apparatus.

[0091] The user provides the response by operating input devices, including a key, switch, dial, mouse, trackball, and joystick, in combination or by doing nothing.

[0092] For example, the user pushes a "function" key and then pushes a "0" key on the apparatus, so that the apparatus acquires information indicating permission to execute all programs. For example, the user selects the character string "maze.cj" by operating the dial on the screen and then pushes a "1" key, so that the apparatus acquires information indicating that a program with identification data "maze.cj" is forbidden from being executed. For example, if the user does not give any specific input regarding permission or forbiddance of the execution of the program before process P209, the apparatus acquires information indicating that the program with the identification data "maze.cj" is forbidden from being executed.

[0093] The response indicating the user's selection may be performed in conjunction with process P207. For example, when the screen displays information shown in Fig. 10, the user then pushes the "0" key on the apparatus, so that the apparatus acquires information indicating permission to execute the program with the identification data "maze.cj".

[0094] In this embodiment, operations of the keys "0", "1", and "function" and the dial, in combination, are described. The selection of keys and the sequence of operating keys described above may be changed. The combination of the input devices described above may also be changed.

[0095] In process P206A, the permission information 108 is read and the determination as to whether the user permits the execution of the program 102 or not is performed.

[0096] This processing is performed by the program interpretation forbidding means 106 shown in Fig. 2.

[0097] In process P204, the program 102 is interpreted.

[0098] This processing is performed by the interpreting means 104 shown in Fig. 2 and is the same as process P104 described above.

[0099] In process P206B, the program 102 is forbidden from being interpreted.

[0100] This processing is performed by the program interpretation forbidding means 106 shown in Fig. 2 and is the same as process P106 described above.

[0101] When process P204 or P206B is finished, the entire processing shown in Fig. 2 is then completed.

[0102] Fig. 3 shows the structure of an apparatus for executing a program according to claim 3.

[0103] Fig. 7 shows a flowchart of processing in the apparatus shown in Fig. 3.

[0104] The structure and processing are described step by step.

[0105] The program 102 and the identifying information 103 received through the external communication channel 101 are the same as those described above.

[0106] The first process P305 in Fig. 7 is performed by the source determining means 105 shown in Fig. 3 and is the same as process P105 above.

[0107] In process P304, the program 102 is executed by the interpreting means 104.

[0108] This process is performed by program interpretation selecting means 111 and the interpreting means 104 shown in Fig. 3.

[0109] In process P310, the program 102 is executed by secondary program-interpreting means 110.

[0110] This process is performed by the program interpretation selecting means 111 and the secondary interpreting means 110 shown in Fig. 3.

[0111] The secondary interpreting means 110 has a mechanism similar to known execution systems included in environments capable of interpreting a program and interprets a program in a manner similar to the known systems. The secondary interpreting means 110 is implemented inside the apparatus as a separate component from the interpreting means 104.

[0112] The program interpretation selecting means 111 is realized by, for example, the following.

[0113] The program interpretation selecting means 111 retains the table illustrated in Fig. 11 therein.

[0114] An entry in this table is made up of identification data of a downloaded program and a corresponding value indicating the status of the program.

[0115] The value indicating the status of the program can take only two possible values: "secure" or "insecure".

[0116] Before the execution of a program, in accordance with the determination by the source determining means 105, the program interpretation selecting means 111 sets the value; if the result indicates that the program is sent from a trusted source, the source determining means 105 records "secure" as the value, and if not, "insecure".

[0117] Before starting interpreting a program, the interpreting means 104 refers to the table shown in Fig. 11 to search for identification data for the program. Interpretation of the program is started only when the value indicating the program status associated with the identification data is "secure".

[0118] If the value indicating the program status is "insecure" or an entry of the identification data is not present in the table, the program is executed by the secondary interpreting means 110.

[0119] In this embodiment, the program interpretation selecting means 111 uses the table, but it may use another structure, such as a tree, a list, or a hash.

[0120] When process P304 or P310 is finished, the entire processing shown in Fig. 3 is then completed.

[0121] Fig. 4 shows the structure of an apparatus for executing a program according to claim 5.

[0122] The block diagram, in which the inquiring means 107 is removed from the structure shown

in Fig. 4, shows an apparatus for executing a program according to claim 4.

[0123] Fig. 8 shows a flowchart of processing in the apparatus shown in Fig. 4.

[0124] The flowchart, in which process P407 is replaced with "performing nothing" in Fig. 8, shows processing in the apparatus according to claim 4.

[0125] The structure and processing are described step by step.

[0126] The program 102 and the identifying information 103 received through the external communication channel 101 are the same as those described above.

[0127] The first process P405 in Fig. 8 is performed by the source determining means 105 shown in Fig. 4 and is the same as process P105 described above.

[0128] In process P407, an inquiry as to whether to permit executing a program is made to a user.

[0129] This process is performed by the inquiring means 107 shown in Fig. 4 and is the same as process P207 described above.

[0130] In process P409, the permission information 108 indicating whether executing a program is permitted or not is acquired.

[0131] This process is performed by the input means 109 shown in Fig. 4 and is the same as process P209 described above.

[0132] In process P406A, the permission information 108 is read and a determination as to whether the user permits the execution of the program 102 or not is performed.

[0133] This processing is performed by the program interpretation forbidding means 106 shown in Fig. 4.

[0134] In process P404, the program 102 is interpreted by the interpreting means 104.

[0135] This process is performed by the interpreting means 104 and program interpretation forbiddance selecting means 112 shown in Fig. 4.

[0136] In process P410, the program 102 is executed by the secondary interpreting means 110.

[0137] This process is performed by the program interpretation forbiddance selecting means 112 and the secondary interpreting means 110 shown in Fig. 4.

[0138] In process P406B, the program 102 is forbidden from being interpreted.

[0139] This process is performed by the program interpretation forbiddance selecting means 112 shown in Fig. 4.

[0140] The program interpretation forbiddance selecting means 112 is realized by, for example, the following.

[0141] The program interpretation forbiddance selecting means 112 retains the table illustrated in Fig. 12 therein. An entry in this table is made up of identification data of a downloaded program and a corresponding value indicating the status of the program.

[0142] The value indicating the status of the program can take only three possible values: "secure", "permitted", or "forbidden". The initial value when the entry is recorded for the first time is set to "secure".

[0143] In process P406A, the program interpretation selecting means 111 reads the permission information 108. If the permission information 108 indicates that the user permits the execution of the program, the program interpretation selecting means 111 records "permitted" as the value indicating the status of the program; if not, it records "forbidden".

[0144] Before starting interpreting the program 102, the interpreting means 104 refers to the table shown in Fig. 11 to search for identification data for the program. Interpretation of the program is

started only when the value indicating the program status is "secure".

[0145] If the value indicating the program status is "permitted", the program 102 is executed by the secondary interpreting means 110.

[0146] If the value indicating the program status is "forbidden" or an entry of the identification data is not present in the table, the program 102 is not interpreted.

[0147] In this embodiment, the program interpretation forbiddance selecting means 112 uses the table, but it may use another structure, such as a tree, a list, or a hash.

[0148] When process P404, P410, or P406B is finished, the entire processing shown in Fig. 4 is then completed.

[0149] The embodiments of the present invention are described above.

[0150]

[Advantages] As described above, according to the present invention, a program and identifying information indicating a source sending the program are received through an external communication channel, and it is determined whether the identifying information indicates a trusted source.

[0151] When the result is that it has not been determined that the program is sent from a trusted source, an execution environment informs a user (using the execution environment) of the result, receives information provided by the user, and determines whether to execute the program in accordance with the information. Alternatively, the execution environment interprets the program by second interpreting means, which is different from the existing interpreting means.

[0152] This arrangement allows an apparatus for executing a program to execute a program having assured security with less resources and at high speed. In addition, the security, confidentiality, and integrity of a downloaded program are assured while maintaining resource savings and low cost.

[Brief Description of the Drawings]

[Fig. 1]

Fig. 1 shows an example of an apparatus for executing a program according to the present invention.

[Fig. 2]

Fig. 2 shows another example of the apparatus for executing a program according to the present invention.

[Fig. 3]

Fig. 3 shows still another example of the apparatus for executing a program according to the present invention.

[Fig. 4]

Fig. 4 shows still another example of the apparatus for executing a program according to the present invention.

[Fig. 5]

Fig. 5 is a flowchart showing an example of the operation of the apparatus for executing a program according to the present invention.

[Fig. 6]

Fig. 6 is a flowchart showing another example of the operation of the apparatus for executing a program according to the present invention.

[Fig. 7]

Fig. 7 is a flowchart showing still another example of the operation of the apparatus for executing a program according to the present invention.

[Fig. 8]

Fig. 8 is a flowchart showing still another example of the operation of the apparatus for executing a program according to the present invention.

[Fig. 9]

Fig. 9 shows an example of program interpretation forbidding means.

[Fig. 10]

Fig. 10 shows a screen displayed by inquiring means.

[Fig. 11]

Fig. 11 shows an example of program interpretation selecting means.

[Fig. 12]

Fig. 12 shows an example of program interpretation forbiddance selecting means.

[Reference Numerals]

101: external communication channel
102: program
103: identifying information
104: interpreting means
105: source determining means
106: program interpretation forbidding means

Fig. 1

101: COMMUNICATION CHANNEL
101: COMMUNICATION CHANNEL
102: PROGRAM
103: IDENTIFYING INFORMATION
104: INTERPRETING MEANS
105: SOURCE DETERMINING MEANS
106: PROGRAM INTERPRETATION FORBIDDING MEANS
A: APPARATUS (I) FOR EXECUTING PROGRAM

Fig. 2

101: COMMUNICATION CHANNEL
101: COMMUNICATION CHANNEL
102: PROGRAM
103: IDENTIFYING INFORMATION
104: INTERPRETING MEANS
105: SOURCE DETERMINING MEANS
106: PROGRAM INTERPRETATION FORBIDDING MEANS
107: INQUIRING MEANS
108: PERMISSION INFORMATION
109: INPUT MEANS
USER
B: APPARATUS (II) FOR EXECUTING PROGRAM

Fig. 3

101: COMMUNICATION CHANNEL
101: COMMUNICATION CHANNEL
102: PROGRAM
103: IDENTIFYING INFORMATION
104: INTERPRETING MEANS
105: SOURCE DETERMINING MEANS
110: SECONDARY INTERPRETING MEANS
111: PROGRAM INTERPRETATION SELECTING MEANS
C: APPARATUS (III) FOR EXECUTING PROGRAM

Fig. 4

101: COMMUNICATION CHANNEL
101: COMMUNICATION CHANNEL
102: PROGRAM

103: IDENTIFYING INFORMATION
104: INTERPRETING MEANS
105: SOURCE DETERMINING MEANS
107: INQUIRING MEANS
108: PERMISSION INFORMATION
109: INPUT MEANS
110: SECONDARY INTERPRETING MEANS
112: INTERPRETATION FORBIDDANCE SELECTING MEANS
USER
D: APPARATUS (IV) FOR EXECUTING PROGRAM

Fig. 5

START
P105: DOES IDENTIFYING INFORMATION INDICATE TRUSTED SOURCE?
P104: INTERPRET PROGRAM
P106: FORBID EXECUTION OF PROGRAM
END
E: PROCEDURE (I) FOR PROGRAM-EXECUTION PROCESSING

Fig. 6

START
P205: DOES IDENTIFYING INFORMATION INDICATE TRUSTED SOURCE?
P204: INTERPRET PROGRAM
P207: INQUIRE OF USER WHETHER TO PERMIT EXECUTION OF PROGRAM
P209: ACQUIRE PERMISSION INFORMATION
P206A: DOES USER PERMIT EXECUTION OF PROGRAM?
P206B: FORBID EXECUTION OF PROGRAM
END
F: PROCEDURE (II) FOR PROGRAM-EXECUTION PROCESSING

Fig. 7

START
P305: DOES IDENTIFYING INFORMATION INDICATE TRUSTED SOURCE?
P304: INTERPRET PROGRAM BY INTERPRETING MEANS 104
P310: INTERPRET PROGRAM BY SECONDARY INTERPRETING MEANS 110
END
G: PROCEDURE (III) FOR PROGRAM-EXECUTION PROCESSING

Fig. 8

START

P405: DOES IDENTIFYING INFORMATION INDICATE TRUSTED SOURCE?

P404: INTERPRET PROGRAM BY INTERPRETING MEANS 104

P407: INQUIRE OF USER WHETHER TO PERMIT EXECUTION OF PROGRAM

P409: ACQUIRE PERMISSION INFORMATION

P406A: DOES USER PERMIT EXECUTION OF PROGRAM?

P406B: FORBID EXECUTION OF PROGRAM

P410: INTERPRET PROGRAM BY SECONDARY INTERPRETING MEANS 110

END

H: PROCEDURE (IV) FOR PROGRAM-EXECUTION PROCESSING

Fig. 9

I1: IDENTIFICATION DATA FOR PROGRAM

I2: STATUS OF PROGRAM

I: EMBODIMENT OF PROGRAM INTERPRETATION FORBIDDING MEANS

Fig. 10

J1: Do you permit the execution of the program "maze.cj"?

0 — Yes

1 — No

J: EMBODIMENT OF INQUIRING MEANS

Fig. 11

K1: IDENTIFICATION DATA FOR PROGRAM

K2: STATUS OF PROGRAM

secured: secure

unsecured: insecure

K: EMBODIMENT OF PROGRAM INTERPRETATION SELECTING MEANS

Fig. 12

L1: IDENTIFICATION DATA FOR PROGRAM

L2: STATUS OF PROGRAM

secured: secure

L: EMBODIMENT OF INTERPRETATION FORBIDDANCE SELECTING MEANS